DIKU NLP Course 2022/2023: Group Project

Last updated September 7, 2022
This project description is likely to be slightly adapted throughout the course. We will notify everyone on Absalon in case of such changes.

The aim of the project is to create a **multilingual question answering** system. The dataset used for this project will be the publicly available TyDi QA[1] dataset, a set of question-document-answer items covering 11 typologically diverse languages. In particular, you will work with the following languages: **English**, **Finnish** and **Japanese**.

The project will build up over the duration of the course, following the in-class lessons. The sections below provide details on each step of the project. We provide recommendations on how to complete each piece of the project, but you may use methods of your choosing, which you learned about in class or outside of class, to complete each task.

The project will be graded as a whole and all parts of this project are compulsory unless noted otherwise. Complete the project in groups of up to **3 students**.[2] No later than **4 November 17:00 Copenhagen time**, submit **one report per group via Digital Exam**, detailing:

- who was responsible for what part of the project;
- motivated explanations for the decisions and considerations you made;
- a human-readable description of what you have implemented; and
- your results and observations.

You also have the opportunity to submit your mid-point progress on the project to receive informal feedback from the course instructors. If you would like to take advantage of this, please submit by 1 October 17:00 Copenhagen time via Absalon. In our experience, this greatly improves the quality of the final submitted projects.

The format of the report should be a PDF document using the ACL template[3] (non-anonymized)[4] no more than 8 pages[5] (but references do not count towards the page limit). The report should be written in English. Your code should be submitted as a .zip file.

When you are ready, submit the report and code on **Digital Exam**. Note that the system will automatically close for submissions at the exact deadline, and that you can update your submission as many times as you would like before the deadline.

[1] https://github.com/google-research-datasets/tydiqa
[2] You can complete the project on your own, but we would not recommend this.
[3] https://github.com/acl-org/acl-style-files
[4] Remove the `[review]` option from `\usepackage[review]{acl}`.
[5] Content after the 8th page will not be graded.

1 Week 36 (5–11 September) Introduction to NLP

1.1 Preprocessing and Dataset Analysis

Before getting started with the implementation, it is useful to familiarise yourself with the dataset. For this purpose, we recommend that you read the TyDi QA paper.[6] Next, download the **Answerable TyDiQA** subtask data.[7] While the **MinSpan** subtask of the original TyDiQA dataset only contains answerable questions, **Answerable TyDiQA** an extension of it that contain also questions that are **not answerable**. Familiarise yourself with the dataset's info card and explore the columns of the dataset. We recommend using Google Colab[8] for free access to computing resources including GPUs.

The next step is to prepare the data for the subsequent steps.

(a) **Implement a preprocessing pipeline that tokenises the instances from each of the languages—English, Japanese and Finnish.** Tokenise inputs at the word level.[9]

(b) Next, to further familiarise yourself with the data, investigate how the questions in the training set normally begin and end. Make an overview of which words are the most common first and last tokens in a question, for each of the three languages (English, Finnish, Japanese). Observe both common question and non-question words that are common as the first and last tokens in a question.

Hint: use a machine translation tool for languages you do not speak.

1.2 Binary Question Classification

For this subtask, you will make the first step towards a Question Answering system for TyDi QA for the three languages used throughout this project. For this assignment, too, *you will use the* ***Answerable TyDiQA*** *subtask data.* **Implement feature-based binary classifiers of your choice, one per language, that predicts whether a question is answerable or not given a context.** That is, the model will predict *"1"* if the answer to the question appears in the context and *"0"* otherwise.

(a) Design classifiers that take only features based on the question, context document, and combinations of the two as input. Motivate your choice of sensible linguistic/lexical features[10] that can be useful for the problem, e.g., bag-of-words, n-gram counts, overlaps of words between question and document, etc.

(b) Train a classifier for each of the three languages, using the training data for that language.

[6]Paper: `https://arxiv.org/abs/2003.05002`, talk: `https://slideslive.com/38929512`
[7]`https://huggingface.co/datasets/copenlu/answerable_tydiqa`
[8]`https://colab.research.google.com/`
[9]Whereas some advanced NLP models use special tokenisation schemes (e.g., BERT using WordPiece Encoding), the Assignments 1 and 2 assume tokenisation on the word level.
[10]Meaning features which are not based on continuous vector representations.

(c) Evaluate the classifiers on the respective validation sets, report and analyse the scores for each language and compare the scores across languages.

2 Week 37 (12–18 September) Representation Learning

The purpose of this task is to familiarise yourself with using different types of representations common in NLP. **Implement an extension to the classifier from Part 1.2, in which you also use features based on continuous vector representations of words/phrases in addition to the already implemented linguistic/lexical features.**

(a) How does performance differ between your solution using word/phrase representations, and your solution without them from Assignment 1.2?

(b) Run experiments in which you use **only** continuous word/phrase representations. How does this affect the performance of your system?

Hint: you will need to ensure that all inputs to your system have the same number of input dimensions, regardless of the length of the input question and document. Hence, if you are using word/phrase embeddings, you will need to condense this into a fixed length. You can combine the representations with the linguistic/lexical features, either at the input level, or deeper in your model.

3 Week 38 (19–25 September) Language modelling

The purpose of this task is to familiarise yourself with neural language models. **Implement an extension to the classifiers from Assignment 1.2 and Assignment 2, in which you instead extract word/sentence representations from neural language models.** You should use monolingual language models; one model per language. You can use any pre-trained monolingual language models, or train or fine-tune neural language models of your own choice.[11]

(a) Select pre-trained language models or train or fine-tune them yourself for the three corresponding languages. If training your own language models, use the subset of **TyDi QA for English, Finnish, and Japanese**. Use the documents and questions from the **MinSpan** training set.[12] You can additionally consider using a larger amount of data than available in the dataset, for instance from Wikipedia, but note that this will likely be quite resource-intensive.

(b) Try sampling from these language models. What kinds of sentences do they generate? What kinds of patterns can you detect in the generated sentences?

[11]You can, for example, find pretrained Transformer language models for different languages, trained with different language objectives, and fine-tuned for different downstream tasks, from Hugging Face, under `https://huggingface.co/models`.

[12]This set can be downloaded from `https://huggingface.co/datasets/tydiqa`.

(c) [**Optional**] If you test multiple language models per language—can you observe any differences in what kinds of sentences they generate?

(d) Evaluate the language model with a commonly-used language model evaluation metric on the TyDi QA validation splits for each language.

(e) Use the final hidden state or a pooled representation from the last hidden layer of the language models as the input to your classifier from Assignment 2. Measure the performance on the TyDi QA validation splits for each language.

Hint: keep in mind that the dataset consists of long documents, so consider how to handle processing this data given the computing and encoding restrictions of RNNs and Transformers. There is more than one potential solution to this.

4 Week 39 (26 September–2 October) Error Analysis and Interpretability

The purpose of this task is to conduct a detailed error analysis and to compare models implemented throughout the course. **Provide insights into the strengths and weaknesses of two chosen models performing the same task based on a detailed error analysis.**

1. Select two of the models implemented in the project. The models should be methodologically different, and achieve different predictive performances, i.e., one having a low performance, while the other having a noticeably higher one. Note that the models have to be performing the same task, e.g., binary question classification. Perform an error analysis comparing the two models. What are the common mistakes of the weak model that the other model predicts correctly? Are there any common mistakes of the stronger model that the weaker model predicts correctly? What are the similarities between the two models?
2. Analyse the better performing model using an explainability tool of your choice. Analyse the tokens from the input that the model considers to be the most important. What are the common patterns associated with each of the classes?
3. Use the identified patterns to write adversarial instances that fool the model to predict the wrong class. Can you see any differences in model predictions on the newly created data?

5 Week 40 (3 October–9 October) Sequence Labelling

The purpose of this task is to move from simple binary question answering to a type of span-based QA, i.e., identifying the span of the paragraph that answers the question. For this, you will use the **Answerable TyDiQA** task data, again for **English, Finnish, and Japanese**, specifically the training and validation sets. **Implement a sequence labeller, which predicts which**

parts (tokens) of a paragraph are likely part of the answer to the corresponding question.

(a) Convert the data to IOB format, labelling each token with an IOB tag depending on whether it is part of the answer or not.

(b) Implement a sequence labeller that takes the question and paragraph as input, and labels each paragraph token with an IOB tag.

(c) Add an extension to the sequence labeller, which uses beam search to select the optimal sequence of labels for the location of the answer in the text.

(d) Analyse how the performance of this system differs with/without beam search and give possible reasons for that.

(e) Perform a qualitative investigation of the predicted answer spans from your model. Do they seem to make sense for the task?

6 Week 41+ (from 10 October) Multilingual QA

The purpose of this task is to implement multilingual QA models based on multilingual text representations. This means the models should not utilise machine translation to convert input text into a common language, but instead can deal with text in different input languages directly. For this task you will again consider the languages English, Finnish, and Japanese. You may use any pre-trained multilingual representations of your choosing.

(a) **Implement the Answerable TyDiQA binary question system from Assignment 3 and the IOB tagging system from Assignment 5 with a multilingual encoder instead of with monolingual ones.**

(b) Perform zero-shot cross-lingual evaluation by training on one language's training set and testing on another language's validation set. In other words, train on English and test on Finnish and Japanese. Then, train on each of Finnish and Japanese and test on English.

(c) Compare the relative performances differences between languages for the two different QA tasks as well as the performance differences between the considered languages with each other.

Hint: the evaluation results for different tasks are not directly comparable as the evaluation metrics differ, but you can compare the relative differences, i.e., if a language that performs well in the binary question answering setting also performs well in the span-based setting.

7 Structure and Grading of the Report

The report should clearly state your group name and the names of all group members. It should describe your approaches for all assignments of this project.

There is no specific structure you have to use, but we recommend having one section per part of the assignment, as well as a section where you state the contributions of each group member.[13] It can also be a good idea to have a conclusions section, where you highlight some of the core challenges, findings and lessons learnt from this project.

While we will verify the submitted code, your project will be mainly graded based on the submitted description document. This also means that we will only assign scores for implementations of methods described in the project. Points will be awarded not only for what you have done, but also for the reasoning behind your decisions. When you describe your choice of a model, a tool, etc., you should provide a brief explanation of it and the reason behind your choice in order to demonstrate your knowledge on the various topics. When you describe the results, you should pick appropriate metrics and baselines for comparison. You should not only report raw numbers, but also attempt to explain result trends and differences between sets of results. If you experimented with different methods for the assignment, it is fine to include the key results in the main part of the report, and additional results for, e.g., ablation studies or unsuccessful early experiments in an appendix. Note that you will also receive overall points for demonstrated mastery according to the criteria listed above.

8 Academic Code of Conduct

You are welcome to discuss the project with other students, but sharing of code is not permitted. Copying code or text from the report directly from other students will be treated as plagiarism. Please refer to the University's plagiarism regulations if in doubt. For questions regarding the project, please ask on the Absalon discussion forum: `https://absalon.instructure.com/courses/61339/discussion_topics`.

In short, plagiarism means copying text or ideas from others without acknowledging the underlying sources. Crucially, this does not mean that you are prohibited from building on others' ideas or use external sources, but rather that you have to properly acknowledge all sources used in your work. This holds for instance for building on code from lectures or lab sessions. If in doubt, we recommend erring on the side of over- rather than under-acknowledging sources.

[13] In case it is not clear which member contributed to which part, all group members will receive the same grade.